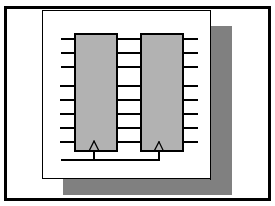
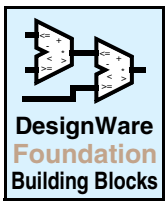




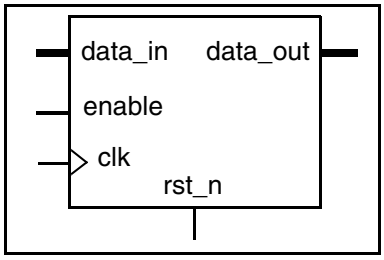
DW_pl_reg

Pipeline Register

Version, STAR, and myDesignWare Subscriptions: IP Directory

Features and Benefits

- Parameter controlled width
- Parameter controlled logic stages
- Parameter controlled input or output register
- Individual enables per register level
- Auto ungroups itself into its parent design for register retiming



Revision History

Description

This component is designed to make it easy to pipeline arbitrary logic or arithmetic structures using the register retiming features of Design Compiler (DC). It contains parameter controlled input and output registers which will stay in place at their respective block boundary - they are not allowed to be moved by DC's register retiming features. The use of both input and output registers at the same time is not allowed (note that input and output register are not available when using DC versions earlier than A-2007.12).

Table 1-1 Pin Description

Pin Name	Width	Direction	Function
clk	1 bit	Input	Clock input
rst_n	1 bit	Input	Active low reset input (see *rst_mode* parameter)
enable	*stages* + *in_reg* + *out_reg* - 1	Input	Bus of enables for each stage of registers
data_in	*width* bits	Input	Input data bus
data_out	*width* bits	Output	Output data bus

Table 1-2 Parameter Description

Parameter	Values	Description
width	> 0 Default: 8	Width of `data_in` and `data_out` buses (and all register between `data_in` and `data_out`)
in_reg	0 or 1 Default: 0	Input register control ■ 0: No input register ■ 1: Input register **

Table 1-2 Parameter Description (Continued)

Parameter	Values	Description
stages	1 to 1024 Default: 4	Controls the number of logic stages in the pipeline.
out_reg	0 or 1 Default: 0	Output register control ■ 0: No output register ■ 1: Output register **
rst_mode	0 or 1 Default: 0	Reset type control ■ 0: Asynchronous reset ■ 1: Synchronous reset
** - The parameters *in_reg* and *out_reg* cannot both be set to 1. In DC versions prior to A-2007.12, input and output registers are not allowed. Thus the value of both in_reg and out_reg parameters must be 0 when using DC versions earlier than A-2007.12		

Table 1-3 Synthesis Implementations

Implementation Name	Function	License Feature Required
rtl	Synthesis model	DesignWare

Table 1-4 Simulation Models

Model	Function
DW03.DW_PL_REG_CFG_SIM	Design unit name for VHDL simulation
dw/dw03/src/DW_pl_reg_sim.vhd	VHDL simulation model source code
dw/sim_ver/DW_pl_reg.v	Verilog simulation model source code

The *stages* parameter refers to the number of logic stages desired after register retiming is performed. The number of register levels is not necessarily the same as the number of logic stages. If no input or output registers are used (*in_reg* = *out_reg* = 0), then there is one fewer register level than logic stages. If either an input register or output register is specified, then the number of register levels is the same as the number of logic stages.

The input bus, enable, provides register enables for each level of registers. The least significant bit (bit 0) of the enable bus provides the enable for the register level closest to the input port, data_in. Conversely, the most significant bit of the enable bus provides the enable for the register closest to the output port, data_out.

Block Diagrams

An instance of DW_pl_reg is placed either at the output or input of a design to be pipelined. DW_pl_reg can be used with DW_lp_pipe_mgr to control the register enables or by the designer's register enabling controls.

Figure 1-1 Block Diagram of DW_pl_reg

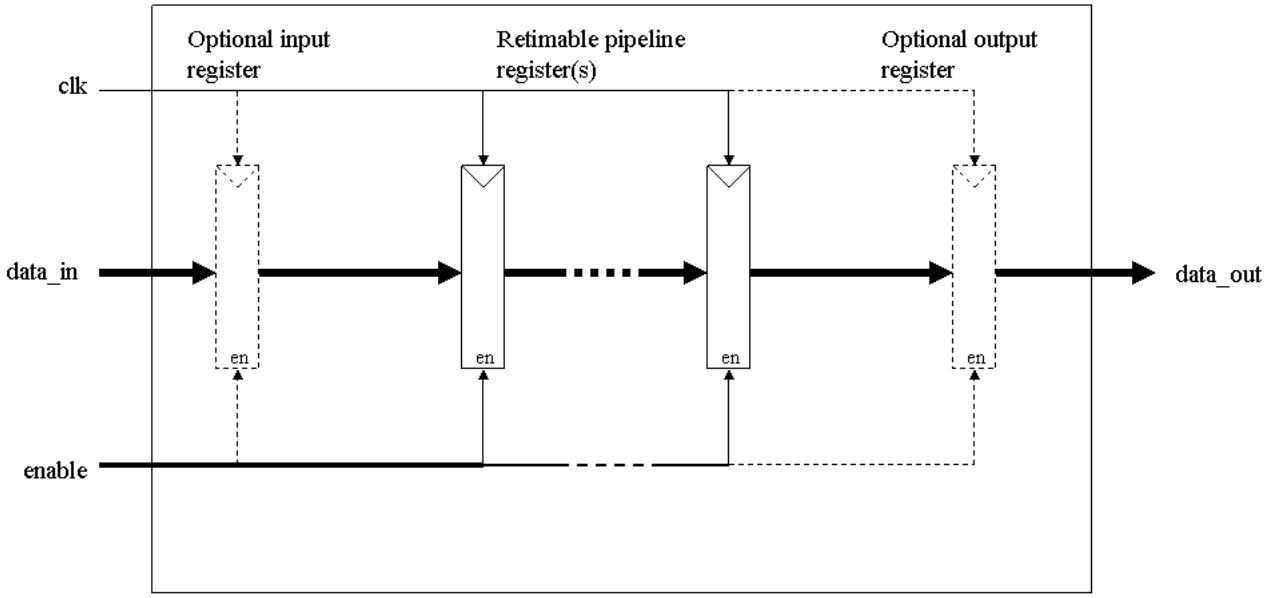


Figure 1-2 Example of DW_pl_reg at Output (Before Compile)

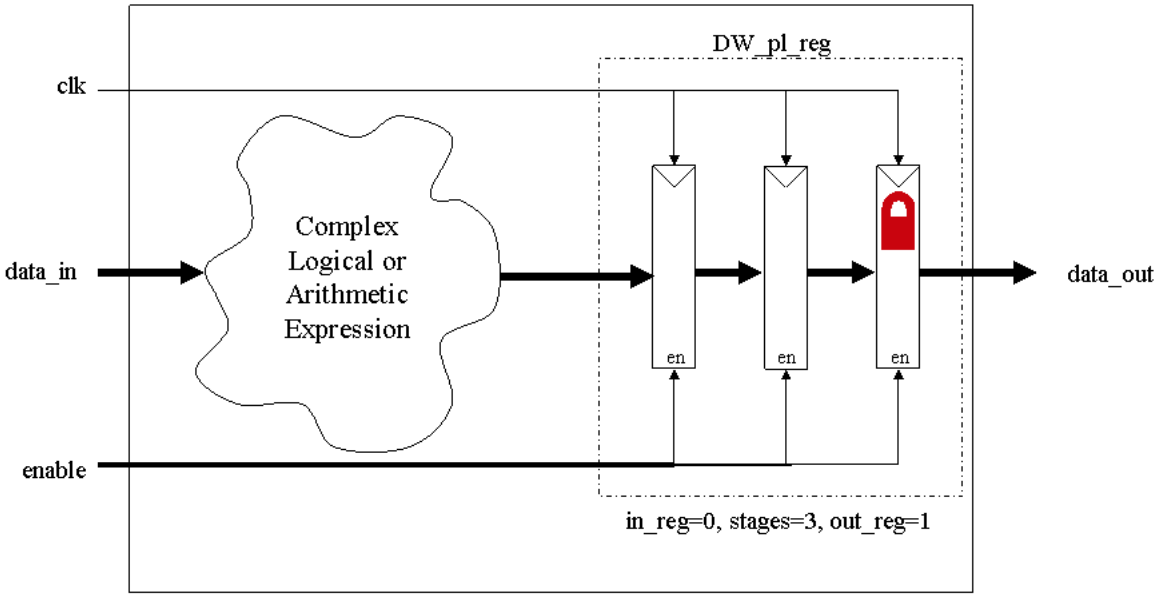


Figure 1-3 Example of DW_pl_reg at Output (After Compile)

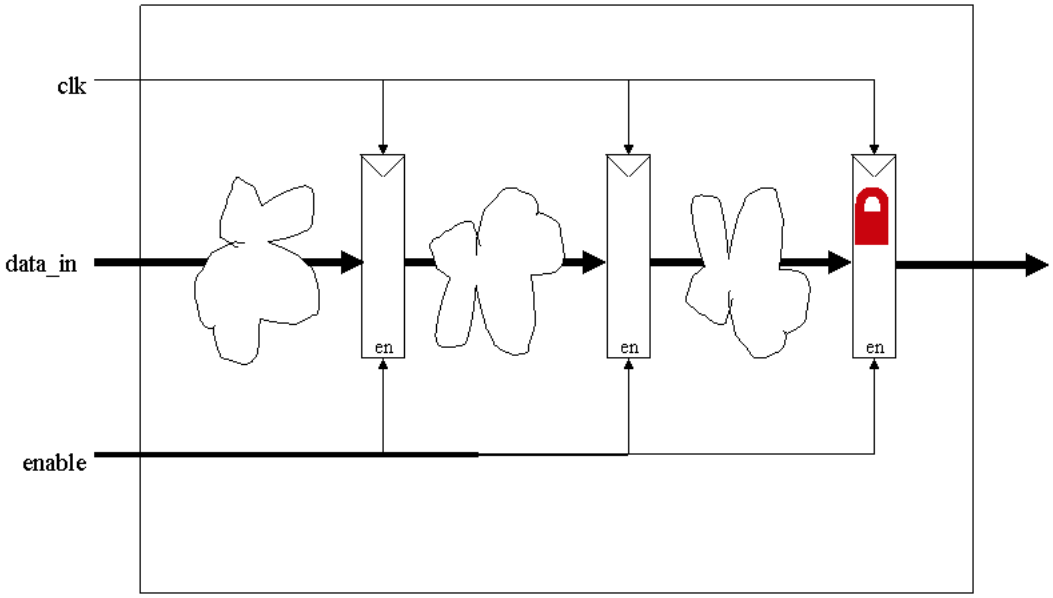


Figure 1-4 Example of DW_pl_reg at Input (Before Compile)

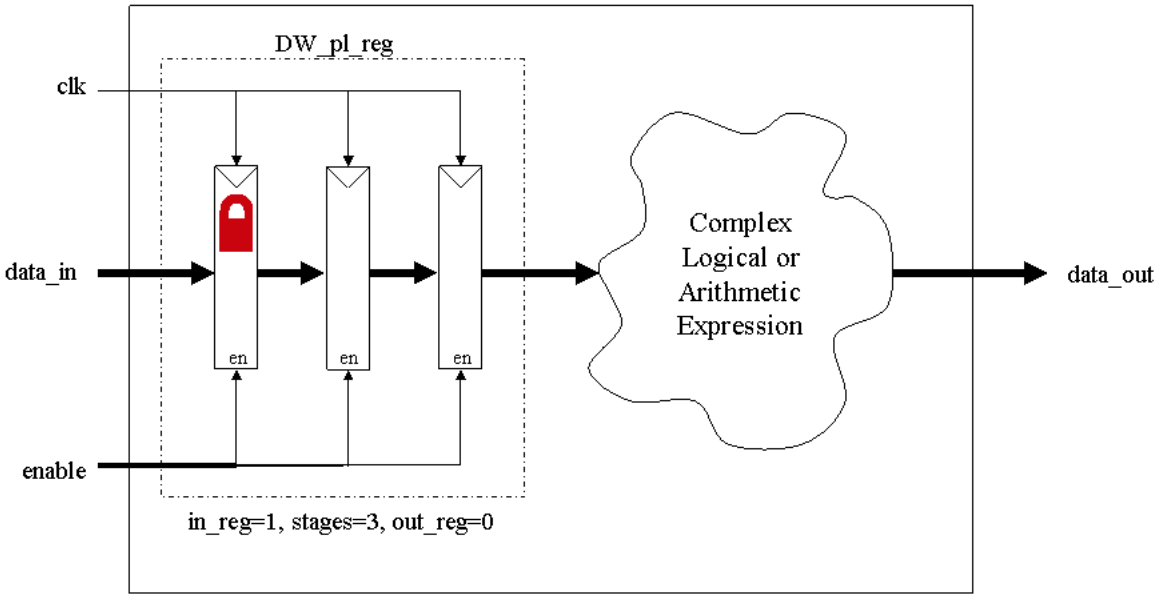


Figure 1-5 Example of DW_pl_reg at Input (After Compile)

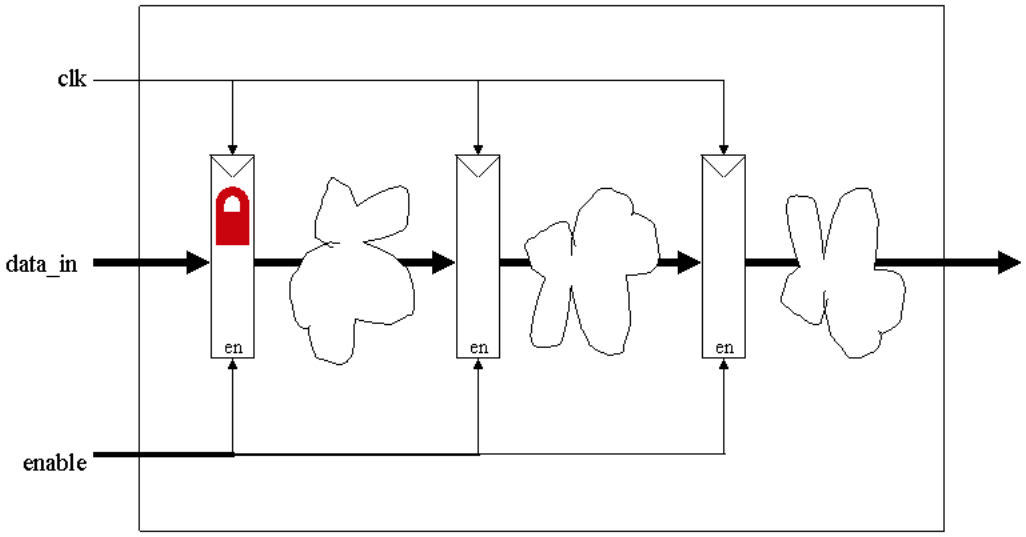


Figure 1-6 Neither in_reg or out_reg are Specified (Before Compile)

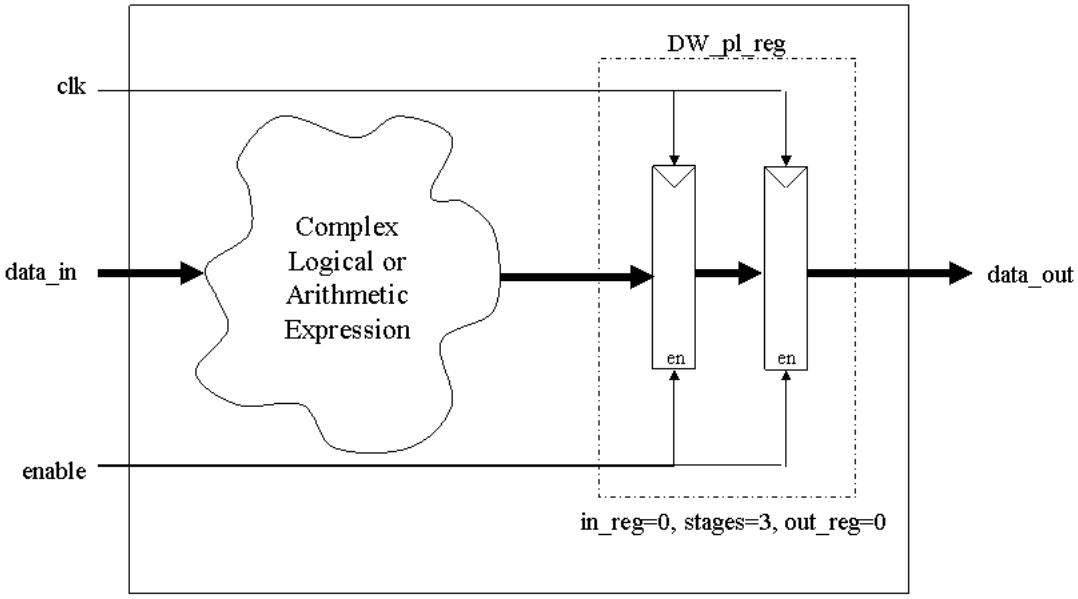
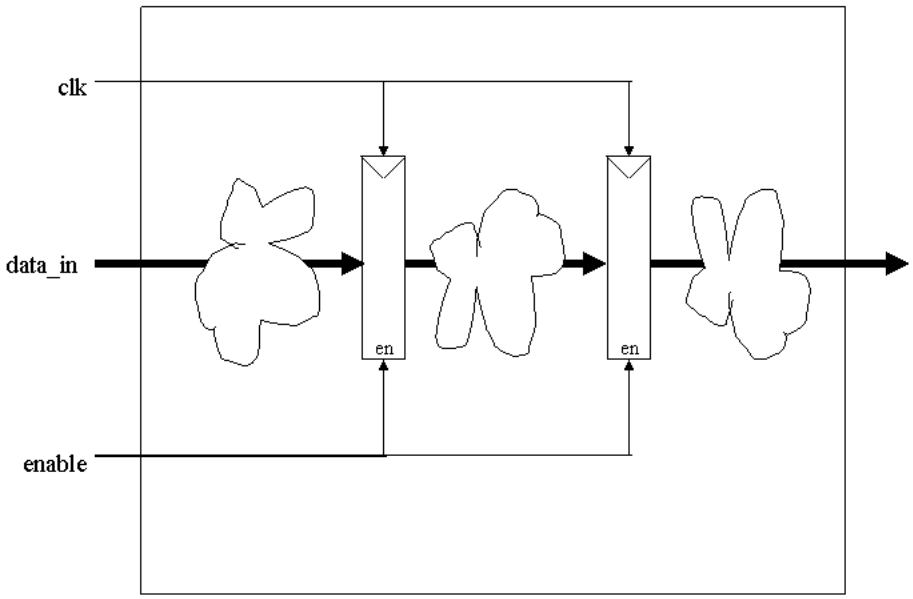


Figure 1-7 Neither in_reg or out_reg are Specified (After Compile)



Suppressing Warning Messages During Verilog Simulation

The Verilog simulation model includes macros that allow you to suppress warning messages during simulation.

To suppress all warning messages for all DWBB components, define the DW_SUPPRESS_WARN macro in either of the following ways:

- Specify the Verilog preprocessing macro in Verilog code:

  ```
  `define DW_SUPPRESS_WARN
  ```

- Or, include a command line option to the simulator, such as:

  `+define+DW_SUPPRESS_WARN` (which is used for the Synopsys VCS simulator)

The warning messages for this model include the following:

- If values other than 1 or 0 are present on a clock port, the following message is displayed:

  ```
  WARNING: <instance_path>.<clock_name>_monitor:
      at time = <timestamp>, Detected unknown value, x, on <clock_name> input.
  ```

  To suppress only this warning message for all DWBB components, use the following macro:

  - Define the DW_DISABLE_CLK_MONITOR macro. You can define this macro in the following ways:
    - Specify the Verilog preprocessing macro in Verilog code:

      ```
      `define DW_DISABLE_CLK_MONITOR
      ```

    - Or, include a command line option to the simulator, such as:

      `+define+DW_DISABLE_CLK_MONITOR` (which is used for the Synopsys VCS simulator)

  This message is also suppressed using the DW_SUPPRESS_WARN macro explained earlier.

Related Topics

- Memory – Registers Overview
- DesignWare Building Block IP User Guide

HDL Usage Through Component Instantiation - VHDL

```
library IEEE,DWARE;
use IEEE.std_logic_1164.all;
use IEEE.numeric_std.all;
use DWARE.DWpackages.all;
use DWARE.DW_Foundation_comp.all;

entity DW_pl_reg_inst is
    generic (
     inst_width : NATURAL := 8;
     inst_in_reg : INTEGER := 0;-- ignored in this design (pipe at output)
     inst_stages : NATURAL := 4;
     inst_out_reg : INTEGER := 0;
     inst_rst_mode : INTEGER := 0
     );
    port (
     inst_clk : in std_logic;
     inst_rst_n : in std_logic;
     inst_enable : in std_logic_vector(inst_stages+inst_out_reg-2 downto 0);
     inst_data_in : in std_logic_vector(inst_width-1 downto 0);
     data_out_inst : out std_logic_vector(inst_width-1 downto 0)
     );
    end DW_pl_reg_inst;

architecture inst of DW_pl_reg_inst is

  signal left_side, right_side : UNSIGNED((inst_width/2)-1 downto 0);
  signal product : std_logic_vector(inst_width-1 downto 0);

begin

  -- split input port, inst_data_in into two equal size multiplier operands
  --
  left_side  <= UNSIGNED(inst_data_in(inst_width-1 downto inst_width/2));
  right_side <= UNSIGNED(inst_data_in((inst_width/2)-1 downto 0));

  -- perform unsigned multiplication
  --
  product <= std_logic_vector(left_side * right_side);

    -- Then, pipeline the module using an instance of DW_pl_reg
    --
```

```
      U1 : DW_pl_reg
      generic map ( width => inst_width,
                    in_reg => 0,
                    stages => inst_stages,
                    out_reg => inst_out_reg,
                    rst_mode => inst_rst_mode )
      port map ( clk => inst_clk,
                 rst_n => inst_rst_n,
                 enable => inst_enable,
                 data_in => product,
                 data_out => data_out_inst );


  end inst;
```

HDL Usage Through Component Instantiation - Verilog

```
module DW_pl_reg_inst( inst_clk, inst_rst_n, inst_enable, inst_data_in, data_out_inst
);

parameter width = 12;
parameter in_reg = 0;// ignored in this design (pipe at output)
parameter stages = 4;
parameter out_reg = 0;
parameter rst_mode = 0;


input inst_clk;
input inst_rst_n;
input [stages+out_reg-2 : 0] inst_enable;
input [width-1 : 0] inst_data_in;
output [width-1 : 0] data_out_inst;

wire [(width/2)-1 : 0] left_side, right_side;
wire [width-1 : 0]     product;

  // split the input bus, inst_data_in into equal size
  // multiply operands
  //
  assign left_side  = inst_data_in[width-2 : width/2];
  assign right_side = inst_data_in[(width/2)-1 : 0];

  // perform the unsigned multiply
  //
  assign product = left_side * right_side;

  // Then, pipeline the module using an instance of DW_pl_reg
  //
  DW_pl_reg #(width, 0, stages, out_reg, rst_mode)
      U1 (
          .clk(inst_clk),
          .rst_n(inst_rst_n),
          .enable(inst_enable),
          .data_in(product),
          .data_out(data_out_inst) );

endmodule
```

Revision History

For notes about this release, see the *DesignWare Building Block IP Release Notes*.

For lists of both known and fixed issues for this component, refer to the STAR report.

For a version of this datasheet with visible change bars, click here.

Date	Release	Updates
July 2020	DWBB_201912.5	■ Adjusted content and title of "Suppressing Warning Messages During Verilog Simulation" on page 6 and added the DW_SUPPRESS_WARN macro
October 2019	DWBB_201903.5	■ Added the "Disabling Clock Monitor Messages" section ■ Added this Revision History table and the document links on this page

Copyright Notice and Proprietary Information

Destination Control Statement

All technical data contained in this publication is subject to the export control laws of the United States of America. Disclosure to nationals of other countries contrary to United States law is prohibited. It is the reader's responsibility to determine the applicable regulations and to comply with them.

Disclaimer

SYNOPSYS, INC., AND ITS LICENSORS MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Trademarks

Synopsys and certain Synopsys product names are trademarks of Synopsys, as set forth at https://www.synopsys.com/company/legal/trademarks-brands.html.

All other product or company names may be trademarks of their respective owners.

Free and Open-Source Software Licensing Notices

If applicable, Free and Open-Source Software (FOSS) licensing notices are available in the product installation.

Third-Party Links

Any links to third-party websites included in this document are for your convenience only. Synopsys does not endorse and is not responsible for such websites and their practices, including privacy practices, availability, and content.

Synopsys, Inc.
www.synopsys.com